LISTING OF THE CLAIMS:

Please **cancel** claims 1-15 without prejudice.

Claims 1-15 (Canceled)

Please **add** the following new claim:

16. (New) A method of implementing a mutual exclusion lock, the mutual exclusion lock capable of preventing at least one acquiring process from operating on at least one shared data object, the at least one acquiring process identified by at least one acquiring process ID, the mutual exclusion lock including at least one variable capable of storing the at least one acquiring process ID, wherein only the at least one acquiring process identified by the at least one acquiring process ID stored in the at least one variable can operate on the at least one shared data object, comprising the steps of:

reading the at least one lock variable in the lock structure;

checking whether the lock structure is busy;

if the lock structure is busy, such that the at least one lock variable contains an old ID of an old process that has previously acquired the lock structure, the acquiring process queries the programming environment whether at least one old process is dead or alive using at least one old ID,

if the old process is alive, the method starts over and the acquiring process reads the at least one lock variable in the lock structure,

if the old process is dead, the acquiring process performs a Compare-and-Swap (CAS) on the at least one lock variable, the at least one old ID and the at least one process ID,

if the CAS fails, the method starts over and the acquiring process reads the at least one lock variable in the lock structure,

if the CAS is successfully performed, such that the at least one old ID is replaced by the at least one process ID,

performing a recovery mechanism for recovering the at least one shared data objects to a consistent state such that the acquiring process can access the at least one shared data objects,

if the lock structure is not busy, such that the at least one variable contains a clear valve, the acquiring process attempts an atomic operation CAS on the at least one lock variable, the clear value and the at least one process ID;

if the CAS is not successfully performed, the method starts over and the acquiring process reads the at least one lock variable in the lock structure;

if the CAS is successfully performed, such that the at least one process ID is written to the at least one lock variable,

operating on one or more shared data objects originally protected by the lock structure;

resetting the acquiring process via the recovery mechanism so that it includes information of the acquiring process accessing the at least one lock variable and is ready for the next process to access the lock structure, wherein the recovery mechanism keeps a log of addresses pointing to the at least one shared data shared objects and values of the at least one shared data objects; and

releasing the lock.